Defense Technical Information Center Compilation Part Notice

ADP014161

TITLE: Multi-Disciplinary Simulation of Vehicle System Dynamics

DISTRIBUTION: Approved for public release, distribution unlimited
Availability: Hard copy only.

This paper is part of the following report:

TITLE: Reduction of Military Vehicle Acquisition Time and Cost through Advanced Modelling and Virtual Simulation [La reduction des couts et des delais d'acquisition des vehicules militaires par la modelisation avancee et la simulation de produit virtuel]

To order the complete compilation report, use: ADA415759

The component part is provided here to allow users access to individually authored sections of proceedings, annals, symposia, etc. However, the component should be considered within the context of the overall compilation report and not as a stand-alone technical report.

The following component part numbers comprise the compilation report:
ADP014142 thru ADP014198

Multi-Disciplinary Simulation of Vehicle System Dynamics

W.R. Krüger, O. Vaculin, W. Kortüm
DLR - German Aerospace Center
Institute for Aeroelastity
D - 82234 Wessling
Germany

Wolf.Krueger@DLR.de
Ondrej.Vaculin@DLR.de
Willi.Kortuem@DLR.de

Summary

Modeling and computer simulation play an important role in all engineering disciplines. As specialized simulation tools have become very sophisticated and, at the same time, the simulation of complex systems and phenomena showed the limits of mono-disciplinary approaches, multi-disciplinary simulation has gained wide acceptance.
For the coupling of different simulation tools interfaces are necessary, including both aspects of physics and numerics as well as of software engineering. This paper tries to give a general classification of interfaces between simulation tools. Following, the multibody simulation approach is presented. With a great number of interfaces to other engineering disciplines like FEA, CAD, CFD, and control design engineering, multibody simulation programs are true multidisciplinary tools which can be used from the pre-design phase to trouble shooting on a production vehicle. As an example, the MBS tool SIMPACK and its integration in the concurrent engineering loop will be presented along with two applications from automotive and aerospace design.

1 Multi-Disciplinary Simulation

Vehicles, i.e. ground vehicles (cars, trucks, trains) as well as air, space and water vehicles, today are complex systems. Requirements of shorter development times, greater safety, longer life time, greater comfort and lower costs have made computer based simulation a necessary tool of the development process. As manufacturers as well as civil and military customers try to incorporate multidisciplinary design methods in the conceptual design phase, a systematic approach needs to be introduced.
Modeling and computer simulation have become tools in all engineering disciplines. Two modeling philosophies for multidisciplinary simulation exist, Fig. 1:

- In one approach, all model components are implemented in a single modeling or simulation tool, using common libraries or a common modeling language, and creating a single model comprising elements of all involved disciplines.
- In a second approach the coupling of specialized tools by the means of interfaces is performed. This is especially suited for systems where sub-models already exist in specialized tools and where those models are too large and complex to be transferred into a single simulation tool.

This paper will deal only with the second approach, i.e. with the coupling of tools via interfaces.

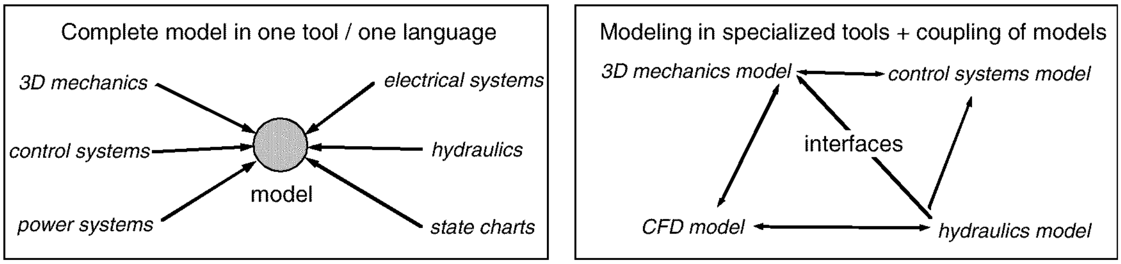


Figure 1: Approaches to multidisciplinary simulation

Paper presented at the RTO AVT Symposium on "Reduction of Military Vehicle Acquisition Time and Cost through Advanced Modelling and Virtual Simulation", held in Paris, France, 22-25 April 2002, and published in RTO-MP-089.

The most widely used computer aided engineering (CAE) tools are computer aided design (CAD), finite element analysis (FEA), control design (often called CACE - Computer Aided Control Engineering), and computational fluid dynamics (CFD). A mediating role between these disciplines is taken by the multibody simulation (MBS) approach. It aims at the simulation of the total vehicle dynamics and offers a good compromise between "fast", "robust", and "exact" simulation [1].
The models used in the engineering fields differ considerably depending on application and the complexity of the task. As an example, in "classical" flight mechanics the aircraft was often represented as a point mass (the coupling of flight mechanical and structural oscillations, of course, today demands a more detailed modeling). Contrary to that, the methods of the finite element analysis and computational fluid dynamics decompose structure and surface of the aircraft in millions of small computational units, a development that has been made possible by the powerful improvement of computer hardware and software in the last decades. In addition, modern CAD programs allow the design of a virtual prototype before a single component is in production. However, this large versatility of models requires an enormous, sometimes redundant modeling effort, and makes it difficult to exchange the obtained results.
Cheap, small and powerful electronics and actuator technology enabled the development of mechanical devices closely interacting with control facilities. For such "mechatronic" systems, an integrated design of mechanical structures and control is indispensable. Multibody simulation is well suited for this procedure and is therefore an important tool in the concurrent engineering process. Multibody simulation allows model simulation and analysis using the know-how of all engineering disciplines mentioned above. To be able to perform these tasks, the program needs intelligent bi-directional interfaces to tools of neighboring disciplines like CAD, FEA, and CACE which allow a continuous comprehensive data exchange. Multibody simulation is suitable both for the pre-design and for the analysis of existing systems, and can be applied for stability and comfort analysis, aircraft response on certain maneuvers, for ground and gust loads, and for life-time prediction. A further advantage for the design process is the possibility to perform parameter studies on a complex simulation model and to optimize free parameters ("design-by-simulation"). Finally, an MBS program is used to calculate system response in a large number of critical operational cases automatically which is of advantage for certification cases. A multibody simulation tool which fulfills these requirements is an essential part of the integrated design process.

2 Interfaces for Coupled Simulation

2.1 Classification of Interfaces

Simulation tools have usually been designed as stand-alone applications in a prescribed work flow. Any two tools rarely use the same native model description or data structure. Interfaces provide a means of communication between two or more coupled applications.
Interfaces are implemented in a variety of ways, and several possibilities for the classification of interfaces exist. When looking at interfaces it is important not only to take into consideration the implementation issues but also their mathematical and physical background. The classifications presented in the following section are therefore based on functionality and work flow, mathematical and physical properties, and software and hardware implementation aspects. It should be noted that a classification cannot always be unambiguous. Other aspects as those mentioned exist, and interfaces can belong to different categories at the same time.

2.2 Functionality / Work Flow

Uni-Directional vs. Bi-Directional Interfaces

Interfaces can be categorized in terms of work flow aspects. Here, a distinction can be made between uni-directional and bi-directional interfaces, Fig. 2. An uni-directional interface is needed if one program is used as a pre-processor for a second program. Typical examples are grid generators for finite element analyses. Bi-directional interfaces handle the flow of information between two running simulations. Typical examples are co-simulation interfaces.

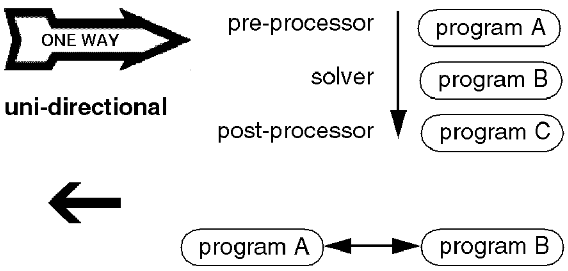


Figure 2: Uni-directional and bi-directional interfaces

2.3 Mathematical / Physical Aspects

Model Description

Depending on application and software, simulation models are described in various ways. For the classification of interfaces it is helpful to distinguish between different model descriptions.

First, simulation models are often described in *application specific parameters*, Fig. 3a. In this case, only the class of a model element, often represented by a number, and values for pre-defined variables are given in an input file. An example is the input file of the MBS code SIMPACK (see Sec. 3) where a certain library element, e.g. a joint type or a force element type is described by its number, and for each element a different set of input values are pre-defined. Other simulation codes, e.g. FEA codes, use the same principle to describe models.

Such a description has the advantage that parameter-based input files are relatively short and of low complexity. However, the parameters in such input files do not give a lot of information about the underlying physical element definition. Interfaces are often based on such native model descriptions, and especially in the case of commercial packages changing the input file is often the simplest way to access these programs in an automated way.

In a second class of models, the so-called *descriptive models,* Fig. 3b, the physical properties of the systems as well as the parameters are defined. This includes particularly models described by differential equations where a solution in time space can only be obtained by the use of an additional solver. In the general case those models can be a function of an arbitrary number of parameters; a special case often used for model exchange are state-space matrices, i.e. linear time independent models. The solvers used for generating solutions for descriptive models depend strongly on the form and numerical properties of the systems.

A third class of models is formed by the so-called *operational models*, Fig. 3c. The output of an operational model is directly the requested response, e.g. in time space. Thus, operational models can either be differential equations with a solver or analytical models where a response can be calculated directly from the input. An operational model can be a 'black box', meaning that the actual model properties are hidden from the user and only well-defined responses on single inputs are given. Therefore operational models are common for interfaces, especially for co-simulation purposes, see Fig. 3.

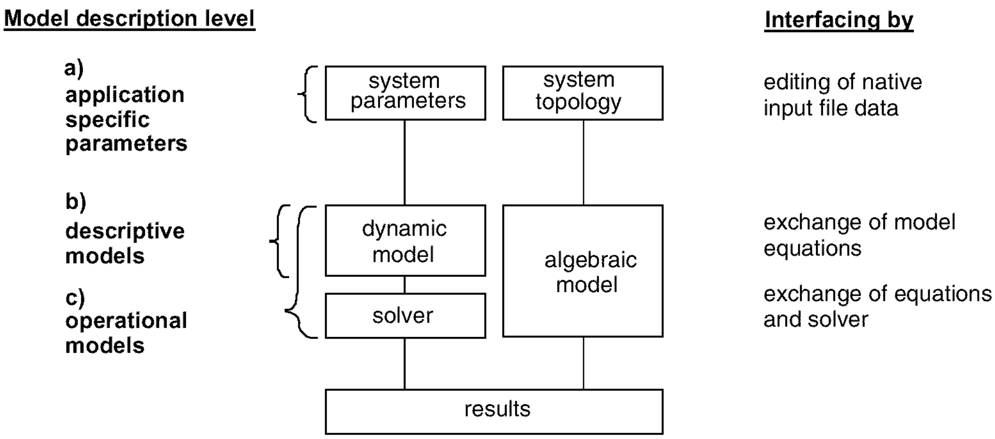


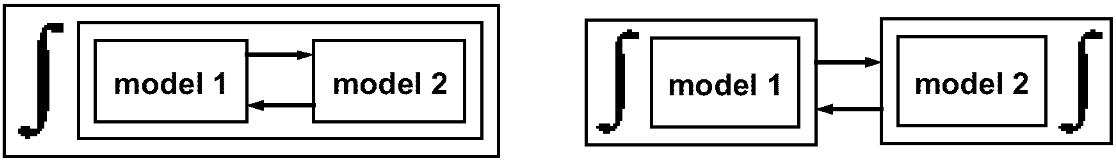
Figure 3: Levels of model description (acc. to [2], modified)

Numerical Integration

Another classification of interfaces is based on their numerical integration schemes. The numerical integration of the coupled system can be performed in one tool by a common numerical integrator; this method is often called *tight* or *close coupling*, Fig. 4a. In this case, the sub-models have to be connected into one complete model and all the states of that model have to be accessible by the numerical integrator. Furthermore, the integrator must be able to handle all types of model behavior and equations used by the included models. The performance of the numeric integration of the coupled system should remain acceptable. Performance and numerical stability can have limits if the numerical properties or the dimensions of the sub-models differ widely.

The numeric integration can also be distributed. In this case the coupled tools use each their own solvers and only inputs and outputs are exchanged, most often at pre-defined communication time points, thus using explicit overall time integration methods. This scheme is often called *weak* or *loose coupling*, Fig. 4b. The states of one sub-model are hidden from the integrators of the other model the disciplines, hence the common name *co-simulation*, the calculation performance can be increased. However, the communication intervals have to be chosen carefully for reasons of performance and stability. Furthermore, it can be shown that some systems, e.g. with closed kinematic loops, do not converge at all with an explicit loose coupling scheme [3].

It should be noted here that both close coupling and loose coupling can be achieved independently from the selected implementation method. However, in practical applications the word co-simulation is often used exclusively for loose coupling in combination with a multi-process or IPC solution. In the following, the term co-simulation will be used as a synonym for loose coupling in general.



a) close coupling **b) loose coupling**

Figure 4: Numerical integration for close and loose coupling

Model Size Adaption

Often models of different complexity are coupled. Differences are either the model size, e.g. the number of degrees of freedom, or in the type of system description. Many physical problems can be described, e.g., dimensionless, in one, two, or three dimensions. If models of different complexity are coupled, solutions have to be found to either reduce the complexity of a sub-model to that of the main model or to interpolate between the sub-models. An example for model reduction are the use of modal representation of flexible bodies or the mathematical model reduction techniques used in control design; an example for the need of interpolation is the simultaneous use of 1D, 2D and 3D models in a turbine simulation.

2.4 Software / Hardware and Implementation Issues

Programming

From the programming implementation point of view the interface can be realized as a *single process* or a *multi-process solution*. This classification is independent of the selected numerical integration aspect. Single processes can be obtained on the source code level or on the object code level. In the first case, source code is transferred and all sub-models or programs are compiled and linked into a single executable. This solution makes the interface platform independent.

On the other hand it is possible to interchange pre-compiled objects and link them into a common executable. This can only be done, however, if all code modules have been compiled for the same platform and operating system. In a multi-process solution all models are simulated in their own executables.

Data Transfer

In a coupled simulation data has to be transferred between the sub-models. Data transfer can be performed inside a code by defined parameter lists of subroutine calls or between codes by file transfer,

inter process communication, or a mixture of both. The choice between the methods depends on the amount of information exchanged, performance, and the simulation environment available.
File interfaces are often used if models are results of pre-processors, have to be portable across platforms, and if a large amount of data has to be transferred between simulations. They are exported from one program and imported by the partner program. Inter process communication (IPC) can be chosen if the processes run in parallel, the amount of data is not too large, and the processes can be connected by a network.
Inter process communication in itself is a large field, and the selection of soft- and hardware is based on the requirements. Communication can be achieved by using directly basic functionalities of operating systems as shared memory or sockets, or by using more comprehensive commercial or public-domain packages which supply communication libraries as PVM, MPI, or CORBA.
When large amounts of data have to be exchanged, e.g. in a coupled simulation of CFD and FEA programs, often file interfaces and IPC are used in parallel. Communication routines are used to schedule the process, but the bulk of the data describing a model is exchanged by files.

Platform Dependence

The coupling of simulations can be realized either on a single CPU *single platform, single node*, several computers of the same type (e.g. clusters) or on different nodes of the same computer *single platform, multi-node*, or on different computers of different types and/or operating systems *multi-node*. All these variations require different solutions for simulation interfaces.
Evidently, a single process solution (see above) as a rule runs on a single node. [1] However, a multi-process solution can, and often will, also be limited to a single node. For this limitation there are a number of advantages. First, often the coupling effort is smaller for non-distributed calculations, because all developments can be made in the same environment, network problems are avoided, and some types of coupling methods (e.g. shared memory) are only accessible this way. Additionally, only one implementation of coupling software is necessary. However, all codes have to be available for the same platform, and questions of available computer memory and computational power for the coupled simulation have to be taken into consideration.
Multi-node solutions of single processor types address this problem by multiplying the power of the hardware while using the same working environment for all nodes. In addition to a single-node solution a scheduling scheme to distribute work load on the nodes is necessary. A multi-node solution is used when many parallel computations of similar structure are required, e.g. in multi-block CFD analyses and in simulations for optimization.
In many cases programs are specialized for different environment, e.g. MBS programs for workstations and CFD programs for high-performance computers. In other cases, programs might be limited to special computers for reasons of (non-)portability or licensing. In these cases, a multi-platform solution has to be achieved. Interfacing routines have to be available for all included platforms, different scheduling systems, e.g. cuing vs. have to be integrated
For complex work flows comprising several programs on distributed networks a number of specialized coupling libraries (e.g. CORBA) and work flow managers, addressing the questions mentioned above, have been developed.

3 MBS as a Basic Element of Interdisciplinary System Dynamics Analysis

3.1 Multibody Systems

Originally, MBS software was designed for the analysis of purely mechanical rigid body systems, sometimes added by force laws from other fields such as hydraulics or electronics, mostly included as source code. Since rigid body MBS is not relying on the exact structure and geometry of its components its main applications were principle dynamic investigations in the early development phase of a project.
Today the request for the features of MBS-software, in particular for vehicle system dynamics, is much more demanding. Modern MBS software packages enable interdisciplinary modeling and analysis, either by own enhancements of the MBS functionality or via interfaces to other CAE tools. As a rule, the individual extensions of MBS programs are well adopted to the needs of MBS computation but limited in their facilities and performance. Interfaces to other CAE software on the other hand not only

[1] Depending on the structure of the program scheduling algorithms might be able to distribute work load even of single process simulations to different nodes of a computer.

offer the entire possibilities and functionality of proven software tools but widely reduce the modeling effort as most of these models already exist anyhow, e.g. for CAD drawings or FEA stress analysis, and only need the appropriate conversion.

3.2 SIMPACK

The MBS program package SIMPACK, [4], has been developed at DLR (German Aerospace Center) as a tool for the simulation of complex dynamic systems in aerospace, transportation (vehicle) systems and robotics applications. Consequent upgrading has matured SIMPACK from a typical MBS-code for analysis of specified systems into a mechatronic simulation and design tool. The basis of SIMPACK are its multibody formalisms, i.e. the algorithms which automatically generate the equations of motion. In SIMPACK CPU-time-saving $O(N)$-algorithms (the number of operations grows only linearly with the number of the degrees-of-freedom) are establishing the equations of motion in explicit or in residual form. The equations of motion of the MBS are set up in the form of ordinary differential equations (ODE) or - particularly in the case of closed-loops - optionally in differential-algebraic form (DAE). Adequate solvers, i.e. numerical integration algorithms, were incorporated or developed, some of them in close correlation with formulating the equations of motion. Beyond the "normal" solvers for time-integration (i.e. the narrow sense of simulation) a variety of special numerical analysis methods, in particular for linear system analysis (linearization, eigenvalues, root locii, frequency response, stochastic analysis in time- and frequency-domain) were modified for their special use in vehicle dynamics and integrated. Moreover, computational procedures for stationary solutions (equilibria, nominal forces) respectively for kinematic analysis were developed and implemented.
SIMPACK has developed and is maintaining bi-directional interfaces to a variety of CAE program packages, Fig. 5. The most important interfaces are presented in the following sections - including elastic bodies from FEA, defining controllers in a CACE environment and improving the dynamics with a multi-objective optimization tool, importing CAD data and coupling of rigid and elastic multibody systems to aerodynamics and CFD calculations.

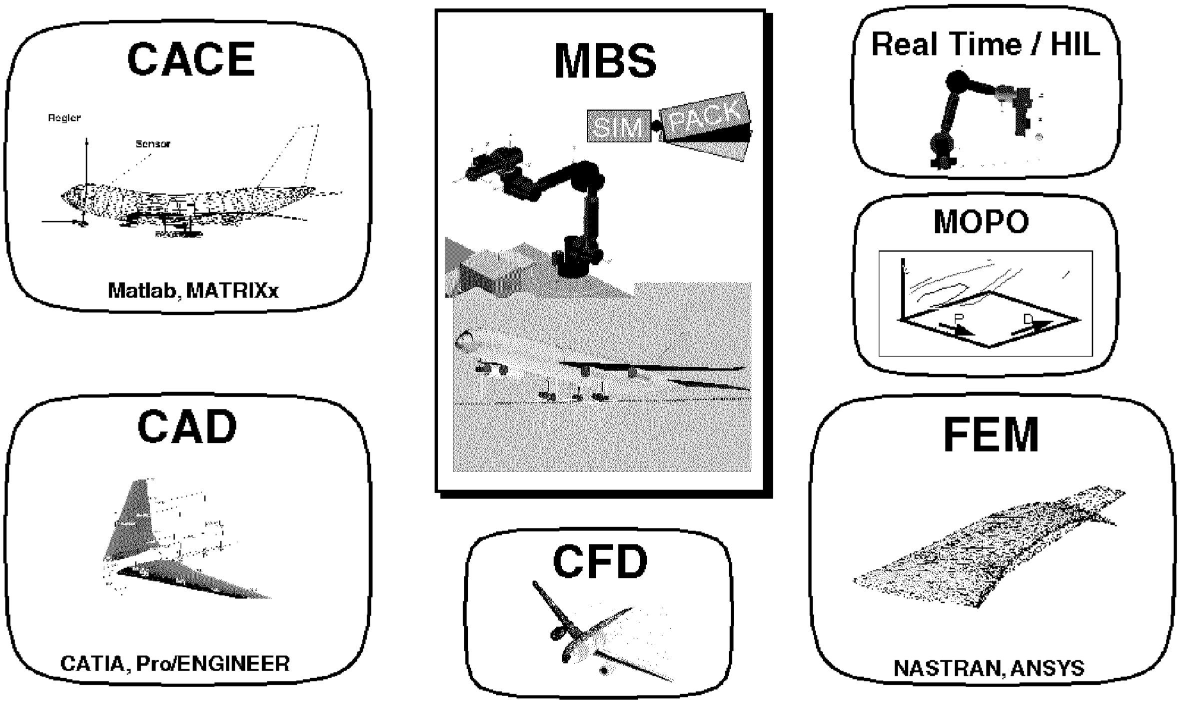


Figure 5: Multibody simulation in the integrated design environment

3.3 Modeling of Elastic Bodies for MBS Analysis

The representation of elastic bodies deriving from FEA modeling in MBS simulation requires adequate pre-processing efforts. A simple transfer of data between FEA and MBS, e.g. for co-simulation, will

result in unacceptable calculation times. In SIMPACK, the pre-processor FEMBS (from FEA to MBS) converts FEA analysis results to an adequate elastic MBS body, Fig. 6.

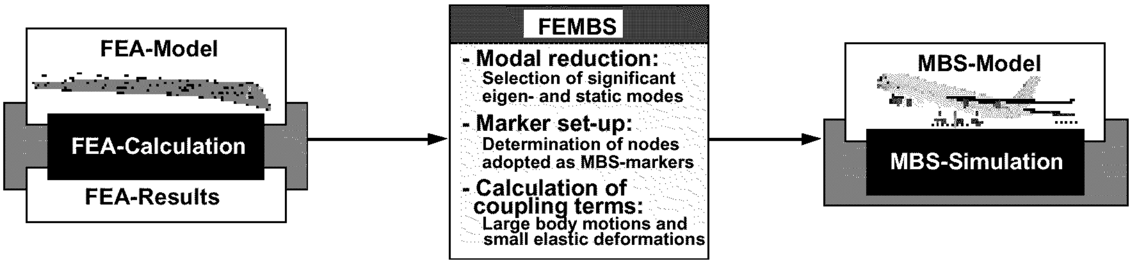


Figure 6: From FEA to MBS: defining elastic bodies in MBS

The spacial motion of an elastic body is divided into a global motion, characterized by the movements of the body reference frame, and its elastic deformation which is expressed by the displacements of all (infinite) body points in relation to the body reference frame, Fig. 7.

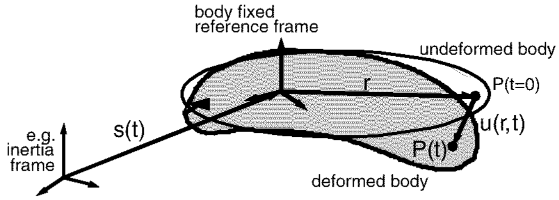


Figure 7: Separation of global motion and deformation

The global motion equals the rigid body motion of a classical rigid MBS body. The location and time dependent body deformation vector *u(r,t)* is split by a separation function often referred to as "RITZ approach" into a location dependent displacement matrix *Φ(r)* and the corresponding time dependent elastic states *q(t)*:

$$u(r,t) = \Phi(r)q(t)$$

The displacement matrix consists of mode shapes of eigenvalue and static load analyses. The eigen- and staticmodes as well as the stiffness matrix are computed in FEA; additionally, geometric stiffening effects, e.g. due to centrifugal forces, can be included. FEMBS enables the user to select only those modes which are necessary to represent the body flexibility for the individual load case. With these data, the equations of motion of the multibody system can be extended:

$$\mathrm{M}(\mathbf{q})\begin{bmatrix}\ddot{\mathrm{s}}_t\\ \ddot{\mathrm{s}}_r\\ \ddot{\mathbf{q}}\end{bmatrix} + \mathrm{k}(\dot{\mathrm{s}}, \mathbf{q}, \dot{\mathbf{q}}) + \begin{bmatrix}0\\ 0\\ \mathbf{Kq}\end{bmatrix} = \mathrm{h}(\mathrm{s}, \mathbf{q}, \ldots),$$

where s denotes the "rigid" MBS states (translational and rotational), k gyroscopic terms, h external forces and K the stiffness matrix. The mass matrix M is enlarged:

$$\mathrm{M} = \begin{bmatrix}\mathrm{mE} & \mathrm{m}\tilde{\mathrm{c}}^{\mathrm{T}}(\mathbf{q}) & \mathbf{C}_t^{\mathrm{T}}(\mathbf{q})\\ \ldots & \mathrm{I}(\mathbf{q}) & \mathbf{C}_r^{\mathrm{T}}(\mathbf{q})\\ \ldots & \ldots & \mathbf{M}_e\end{bmatrix}.$$

The body mass matrix *mE* remains unchanged while the inertia matrix *I* and the STEINER terms mc^T are now dependent from the deformation. The additional "elastic terms" are volume integrals of matrices deriving from *M, K*, Φ and are approximated by TAYLOR expansions.

This so-called "close" coupling of FEA and MBS enables the user to calculate the elastic deformation of flexible bodies fast, in good accuracy and in a form which harmonizes with SIMPACK's MBS-optimized numerical integrators. For a more detailed explanation we refer to [1], [5].

3.4 Control System Design and Analysis

A number of interfaces between SIMPACK and Computer Aided Control Engineering (CACE) software, most notably to MATLAB and MATRIXx, are well established [6].

MATLAB and MATRIXx are tools for control design and system analysis which form a design chain with their block-oriented simulation environments MATLAB-Simulink and MATRIXx-SystemBuild and the code generation tools Real-Time Workshop (MATLAB) and AutoCode (MATRIXx). The packages are similar in structure and complexity which is no coincidence since both programs evolved from the same roots, the original Matlab by Little and Moler (cf. [7]). The tools offer analysis methods in the time and the frequency domain as well as many basic control design functions. They offer different interfaces for model import and export; the interfaces between SIMPACK and MATLAB or MATRIXx are called SIMAT and SIMAX, respectively.

Model Transfer from SIMPACK to CACE

Linear System Interface

SIMPACK models can be linearized and exported in the form of linear system matrices in a MATLAB / MATRIXx-readable format. The model is represented in the following form:

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

where x can consist of rigid-body motion states, states of elastic bodies (in modal formulation, see Sec. 3.3), and states of force elements; the input u can be any kind of excitation, prescribed motion or external force. The models can be the basis for linear system analysis and for control design in MATLAB / MATRIXx. Inside Simulink or SystemBuild the model can be used directly in a state-space block. The interface allows a very fast model export, is platform independent and universal. Restrictions are, as the name suggests, the limitation to linearized models and the one-way data transfer of the MBS environment to the CACE program. The Linear System Interface is an example of a uni-directional interface for close coupling of the systems.

Symbolic Code Interface

Models with non-negligible nonlinear effects can also be exported from SIMPACK in a platform independent way in the form of so-called Symbolic Code. While generally the Symbolic Code is capable of exporting any kind of mechanical system, only models described by ordinary differential equations (ODEs) can be used by the SIMAX Symbolic Code Interface. Here, the model has the following form:

$$\dot{x} = f(x, u, t)$$
$$y = f(x, u, t)$$

SIMPACK generates model dependent, portable FORTRAN code which can be connected to Simulink as an S-function or to SystemBuild as a UserCode Block. With a suitable converter the symbolic code can also be transferred into C to be used in a Hardware-in-the-Loop environment. However, the code is model dependent, i.e. if the multibody system is modified, the FORTRAN code must be generated, compiled, and linked again. Furthermore, no re-transfer of simulation results into SIMPACK is possible. The Symbolic Code Interface, too, a uni-directional interface for close coupling of the systems.

Communication between SIMPACK and CACE

Function Call Interface

The maximum communication between SIMPACK and the CACE tool can be reached by the use of the Function Call Interface which allows to include SIMPACK in MATLAB or MATRIXx in its full functionality. As a bi-directional interface, it also works using the S-function / UserCode Block, forming one simulation module from MATLAB / MATRIXx and SIMPACK routines. The numerical integration is performed in the CACE tool which calls SIMPACK subroutines for the right-hand-side of the equations of motion (close coupling). The interface is restricted to models which can be described by ordinary differential equations. While in MATLAB / MATRIXx only the elements selected for the y-vector as defined in that equation are visible, all the results of the simulation, including the graphical animation of the multibody system, can afterwards be plotted and animated in SIMPACK. It has to be noted, however, that for the Function Call Interface both the CACE tool and SIMPACK have to be available on the same platform since a common executable is formed. Furthermore, for large systems, the integration might become slow when compared to a simulation purely inside SIMPACK because the MATLAB and MATRIXx integrators are not optimized for the solution of mechanical models.

Co-Simulation Interface
If SIMPACK and MATLAB or MATRIXx are available on different platforms, a combined simulation can be performed using co-simulation via inter-process communication (IPC). In that case, each package forms its own executable which communicate by the means of sockets, i.e. a network link providing a two-way communication channel between processes, either user-programmed or based on commercial or public-domain IPC libraries. Data exchange is performed in discrete time steps. Since all MBS model components are solved inside SIMPACK, taking advantage of the optimized integrators, no restrictions to modeling apply. The interface is capable of using models in the differential algebraic equation formulation (DAE):

$$0 = f(\dot{x}, x, u)$$
$$y = f(\dot{x}, x, u)$$

where x includes rigid body states, elastic body states, force element states, holonomic constraints and other algebraic equations to determine additional auxiliary conditions (e.g. for the on-line determination of accelerations and of friction forces). As in the Function Call Interface all simulation results are available for post-processing in both MATLAB / MATRIXx and SIMPACK. Restrictions are a longer simulation time since due to sequential ("step-by-step") co-simulation stability can often only be reached by very small communication intervals.

Transfer of Systems from CACE to SIMPACK

All the interfaces described above can be used to make an MBS model available for control design tools. However, once a control structure is established, it is essential that the complete model can be simulated in the MBS environment for evaluation and optimization purposes. For this reason, two ways have been developed to export a defined control loop from the CACE tool to SIMPACK.

Inverse Symbolic Code Interface
After a control design concept is set up in Simulink / SystemBuild, any chosen parameters can be defined as free parameters and the control structure can be exported. For this kind of model export, MATLAB offers the Real-Time Workshop, MATRIXx the module "AutoCode" which generate portable C code from block diagram models. The code can be used as a user-defined control force element and connected to the multibody simulation via the SIMPACK programmable interface (see Sec. 3.8). However, the Real-Time Workshop and AutoCode are separately licensed which can lead to considerable additional costs.

MBS Syntax Interface
Sometimes not all elements defined in the block diagrams can be exported as source code. Furthermore, sometimes the result of a MATLAB or MATRIXx calculation is only a gain matrix for which the code export would be too cumbersome. In this case it is possible to save the results of the control design in the syntax of single SIMPACK force elements. An element thus defined is then placed in the data base from which the simulation model is assembled, a process which has been automated by the development of special MATLAB and MATRIXx script files.

3.5 Parameter Variation and Multi-objective Optimization (MOPS)

In its analysis toolbox, SIMPACK offers an automized parameter variation which is used to get an overview over system performance as a function of parameter changes. The user defined set of "free" parameters may include mechanical values or force law coefficients as well as control parameters.

The free parameter set cannot only be investigated for its sensitivities: the optimal system configuration can be found with SIMPACK's multi-objective optimization tool MOPS (Multi-Objective Parameter Synthesis), [8]. MOPS is an independent optimization tool whose complete functionality can be operated with the SIMPACK GUI (graphical user interface). The simulation evaluated within the optimization loop can include static, linear and nonlinear simulations of multiple MBS-models, characterized by the same free parameters (Multi Model Optimization). A data handling module is added for structured control of the interactive optimization design process, where design parameters, model and simulation scenarios are varied, starting out from the first optimization test to the final optimal system.

The free system-parameters p_i varied within their given limits until an “optimal compromise” is found. In doing so the criteria $c_i(p)$ (performance indices) are weighted by user-defined factors d_i and the optimization strategy tries to minimize $c_i(p)/d_i$ working always at the (present) $\max_i(c_i(p)/d_i)$, i.e.

$$\alpha^* = \min_p \max_i \left(\frac{c_i(p)}{d_i}\right)$$

with α^* denoting the maximum preference function. The optimum is always a point (depending on d_i) on the PARETO-optimal boundary, [9].

3.6 CAD-Interface

Computer Aided Design (CAD) systems are a central part of the design process. Modern CAD systems allow the definition of 3D models from single parts to complex virtual prototypes. The dynamic analysis of such a system can be done either by including a dynamic solver in the CAD program or by importing CAD data in an MBS program. Both ways have been implemented in SIMPACK. Models from CATIA, Pro/ENGINEER and I-DEAS can be imported in SIMPACK, i.e. the geometric and mass properties as well as the 3D visualization are taken from the original CAD model. SIMPACK can also be completely included as a module in Pro/ENGINEER where all the functionalities are available in the CAD environment and a consistent data handling between CAD and dynamic simulation is achieved.

3.7 CFD-Interface

Aerodynamic forces on rigid bodies can be included in the MBS calculation by simple force elements, e.g. using standard methods of aerodynamic derivatives. Distributed aerodynamic forces on elastic bodies can be included as close coupling, i.e. by introduction of the aeroelastic equations in the MBS model, or by loose coupling, i.e. co-simulation between the MBS code and a computational fluid dynamics (CFD) code.

The *close coupling* enhances the non-linear equation of motion of the hybrid multibody system by “aerodynamic terms”, which are computed in a pre-processing step to the multibody simulation. The approach allows to consider the effects of time-dependent, distributed airloads on the flexible aircraft structure without affecting the simulation efficiency. The pre-processing software tool AeroFEMBS uses standard codes, e.g. three-dimensional potential flow methods, to analyze the aerodynamic loading of the aircraft in high-lift configuration and prepares the corresponding aerodynamics input file for the multibody simulation. Thus, AeroFEMBS allows to include realistic aerodynamic effects into multibody simulation, including distributed airloads on elastic lifting bodies, effects of elastic body deformation on the aerodynamic load distribution, consideration of control surface deflections, velocity-dependent unsteady airloads resulting from elastic deformation and ground effects [10].

For the loose coupling a co-simulation between SIMPACK and a CFD code is established. In general, the MBS representation of the elastic structures and the CFD representation of the surface use different grids. Main points of interest are therefore the data transfer between the codes and the interpolation between the different grids. In recent applications the MPI-based coupling library MpCCI (Mesh-based parallel Code Coupling Interface, [11]) has been used both for data transfer and for grid interpolation [12].

3.8 Programming Interface

A much-used interface for the development of user-defined modules and coupling of external code is the so-called User-Force-Element, a programming interface which allows the introduction of FORTRAN and C elements into the SIMPACK simulation. Various sub-systems can be included, among others controllers (as done in the case of the SIMAT “Inverse Symbolic Code” interface), actuators, tyre models, hydraulic elements. The CFD coupling with MpCCI has also been established by using the programming interface.

3.9 Co-Simulation Interface

In addition to the programming interface SIMPACK offers a standard, open IPC co-simulation interface. The co-simulation is the basis for one of the SIMAT and SIMAX interfaces, another established link exists to the hydraulic simulation tool AMEsim. However, any code which meets the data definition for the interface can be coupled, using either built-in communication libraries or those supplied by the user.

4 Application Examples

4.1 Control Design and Optimization: Semi-Active Truck Suspension for Reduction of Ground Loads

Road Friendly Suspension Design

The following example presents the use of the interface between SIMPACK and MATLAB for the design of semi-active suspensions of trucks in order to fulfil the conflicting demands of road friendliness and comfort. The spatial decomposition approach is used to design the suspension structure. The contribution of the semi-active suspension is verified by experiments including cornering and braking with a complex truck simulation model.
The maintenance of the road networks becomes a very expensive task for road repair authorities. Furthermore, damaged roads cause additional deterioration of the vehicles and also of the road. Controlled automotive suspensions, in particular active and semi-active damping, have been studied for a long time particularly with the comfort objectives. The influence of semi-active suspensions to the decrease of road loads has been already proven by many simulation and field experiments, [13].
The vehicle generated road damage is caused by static and dynamic forces between road and tyre, [14]. Estimations indicate that a fully loaded truck deteriorates a road in the order of magnitude of 10^4 times more compared to the passage of a passenger car. Since presently only the static values of the road-tyre forces are regulated by national and international standards, further investigations must be focused to the reduction of the dynamic forces. The dynamic loads can be reduced by proper suspension design, but passive design has been almost driven to its limits.
The increasing demands on suspensions together with relatively cheap and powerful electronics and actuator technology enable a wide investigation of mechatronic suspensions, active or semi-active, with suitable, road friendly oriented, control laws.

Truck Simulation Model

The simulation model of a heavy duty platform truck is used in this study. The vehicle is designed for long-distance transport on hard-surface roads.
The model is equipped with a steerable front axle and a steering mechanism including a simple driver model. Furthermore, the vehicle brakes as well as a drive train with a velocity controller are implemented. The cabin is fully suspended.
The vehicle model has two basic versions, *passive* and *controllable*. The controllable version is equipped with semi-active dampers Mannesmann Sachs CDC N 50/55 on the axles and semi-active dampers in the cabin suspension. The total mass of the model is 16 tons. The MBS model of the truck consists of 41 bodies and has 64 states in the passive version. 36 outputs are defined for control and evaluation purposes.
For the design of the controller the SIMAT co-simulation interface is used. The SIMPACK solver numerically integrates the mechanical part of the model while the control part is integrated in the MATLAB/Simulink environment. The co-simulation sampling frequency is set to 200 Hz. This approach enables to apply models with closed kinematic loops. Fig. 8 is a 3D representation of the SIMPACK model.

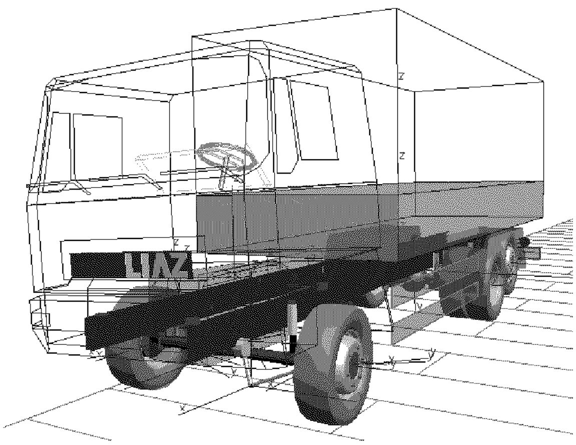


Figure 8: Truck Model

Semi-active Suspension Controller Design

The spatial decomposition approach resolves the contradictory demands on the controller by the *structure of the system*, thus this method can be also called more generally *spatial distribution of control tasks*. The controlled system is divided into two or more subsystems with different objectives for each control. Each system is then optimized using less conflicting criteria or even only one criterion for one specific task. Certainly, such decomposition is not possible to be applied to all dynamic systems.

The suspension systems of a heavy duty vehicle consist of independent suspensions, such as axle suspensions, which are mounted between the vehicle axles and the frame, and suspensions of the vehicle cabin, which support the cabin on the frame. The vehicle in Fig. 8 has four independent suspension systems: (i) front axle suspension, (ii) rear axle suspension, (iii) cabin suspension, and (iv) seat suspension. The strategy is to optimize the axle suspensions exclusively for the objective of road friendliness (minimization of road-tire force fluctuation) and the cabin and seat suspension for the ride comfort.

The controller parameters have been designed by simulation with a Multilevel Coordinate Search global optimization algorithm. The controller parameters are optimized subsequently. The axle controllers are optimized in the first sub-process and then the axle controller parameters are fixed and the comfort controllers are optimized. The semi-active dampers are controlled by well established control laws. The axle suspensions, which are designed as road-friendly oriented, are controlled by the extended ground hook concept. The comfort oriented cabin suspensions are controlled by the skyhook law. The strategy of the controllers and of the optimization is described in detail in [16].

The optimization of the axle suspension for minimization of road-tire force fluctuation can result in an increase of vertical acceleration of the heavy duty vehicle. However, while there are some limits for acceleration transferred to the vehicle cargo, those limits are expected to be fulfilled. A resulting deterioration of the comfort for the driver is solved by optimizing the cabin suspension and the suspension of the seat.

The spatial decomposition method can be applied to active, semi-active and passive suspension systems. Nevertheless, controllable suspension systems can profit from the decomposition significantly more.

Simulation Results

The simulation scenario consists of an S-shaped track with a curve radius of 120 m. The vehicle is excited before the curve entrance by a deterministic ramp. The ramp is 0.08 m high and 5.8 m long, ascent and descent lengths are 2.5 m. The vehicle intensively brakes between 3.5 s and 7.5 s from a speed of 22 m/s down to 4 m/s. The total simulation time is 12 seconds and the initial vehicle velocity is set to 80 km/h.

The simulation results for the truck equipped with a passive suspension compared to the semi-active system are presented in Fig. 9. The peaks at time 0.5 s are caused by the ramp.

Fig. 9a presents the vertical acceleration of the driver. The comfort increase is very significant in this case. The passive suspension of the cabin is relatively soft and moreover the vehicle frame, which is the base for the cabin suspension is better suspended.

The road-tyre forces are presented for the outer right tires of the rear axle in Fig. 9b. The influence of braking is visible at the time 4 s. A reduction of road-tire forces is observable.

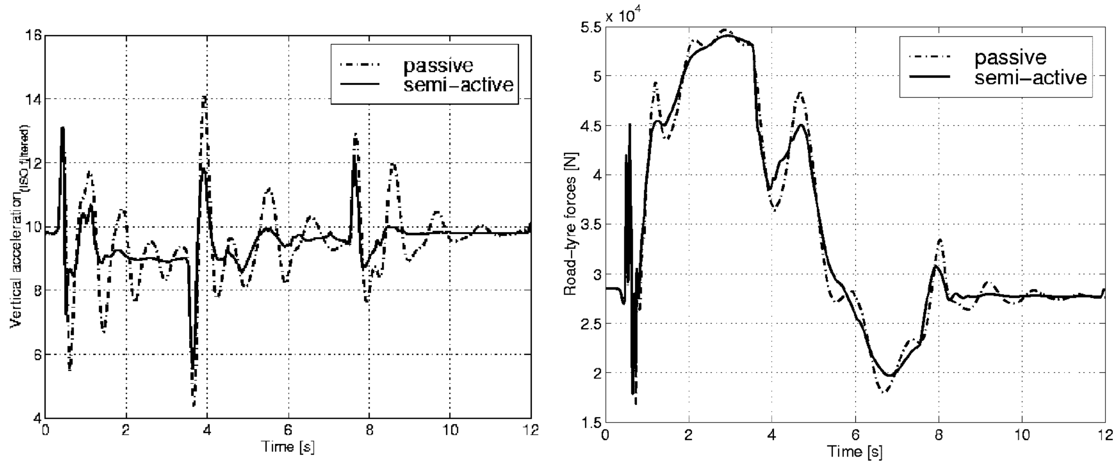


Figure 9: Cabin acceleration and road tire forces of truck

The simulation results indicate that the vehicle equipped with a semi-active suspension designed with the spatial decomposition control profits from a decrease of vertical acceleration and vertical road-tire forces. Generally, the simulation experiments have proved the potential of improvement of road friendliness by semi-active damping for vehicle maneuvers.

4.2 Interaction of Aircraft and Landing Gear

Conventional Design Process

A landing gear of a large transport aircraft has to accomplish a variety of functions. Among others, it has to:

- dissipate the energy of the vertical velocity at touch-down,
- suspend bumps from uneven runways/taxiways and provide a satisfying rolling comfort.

Modern transport aircraft are complex constructions. Their development requires a high level of special knowledge and experience on a multitude of disciplines. Therefore, many aircraft manufacturers confide a specialist with design and fabrication of the landing gear. As a rule, a basic design data set is defined at an early development stage containing, for example, mass and geometry data, configuration, specifications and operating envelopes. Provided with these basic data, the landing gear manufacturer designs a landing gear appropriate for the defined aircraft. Parallel to the work on the landing gear, the "airframer" develops and manufactures the airframe according to the ground load cases set up in cooperation of both partners. The limited knowledge about the partner's part often leeds to calculation and certification cases where simple models represent the other component, e.g. the "drop-test"-model with an equivalent single mass representing the airframe for the design of the landing gear shock absorber. This parallel strategy has the advantage that the development process can rely on fixed data; iteration loops caused by alterations of the basic design data due to optimization results of the partner are avoided.

On the other hand, the dynamic behavior of the combined system which is, in case of high structural flexibility, decisively influenced by dynamical interactions cannot be evaluated. A simple example shall illustrate this effect [17]: A large transport aircraft is rolling, at high velocity, over two sinusoidal bumps of a height of 3.8cm (1.5in), 21m (70ft) apart, Fig. 10. If this case is calculated with a conventional rigid airframe model, we receive a result for the applied vertical acceleration in the cockpit of 0.2g, Fig. 10, curve 1. Including the elasticity of the airframe, the maximum acceleration of the cockpit triples because of resonance phenomena, Fig. 10, curve 2.

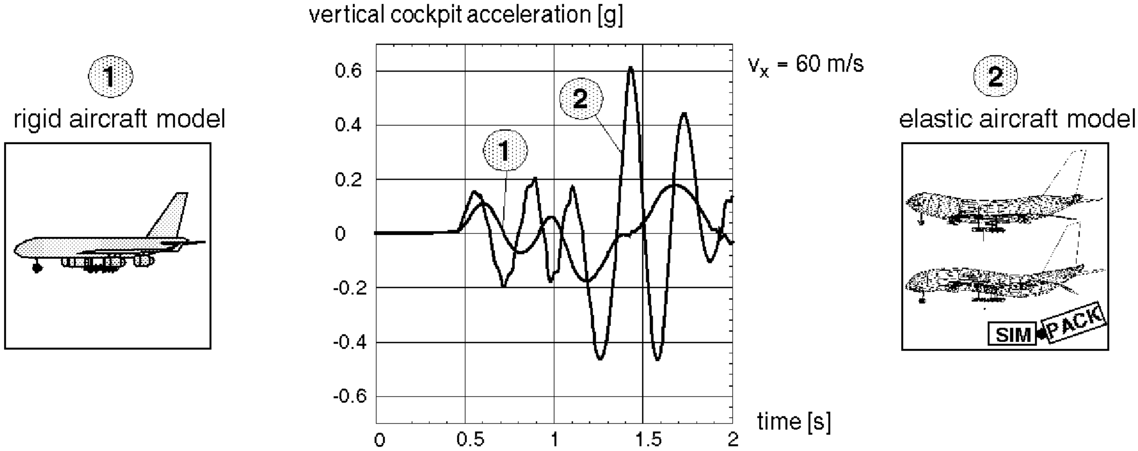


Figure 10: Comparison Between Rigid and Elastic Modeling

Integrated design

The later problematic interaction effects between airframe and landing gear are detected, the smaller is the available design freedom for alternative solutions - with progressively climbing costs. Preventive efforts should therefore take effect as early as possible to implement necessary adoptions to the design quickly, efficiently and without loss of performance.

Consideration of component interactions due to structural elasticities influences the entire design process. Neither airframer nor landing gear manufacturer can rely on a fixed set of basic design data; the level of detail of the basic design data set necessary to assess the most important influences on the system dynamics enforce the inclusion of data which are subject to permanent changes during the development process. This requires alterations on the level of design strategy, e.g distribution of responsibility or data handling, as well as modifications of the computational methods.

Compared to the conventional design strategy where the manufacturers develop their product in a widely independent development process, Fig. 11, left, the so-called “Integrated Design” strategy leads to a close cooperation [18]. The final solution matures out of the development and optimization process of both manufacturers, Fig. 11, right. Unfavorable lay-outs are detected early and can be corrected with comparatively small effort.

Basis for such a coupled design process is an aircraft/landing gear model comprising data from both the airframer and the landing gear manufacturer.

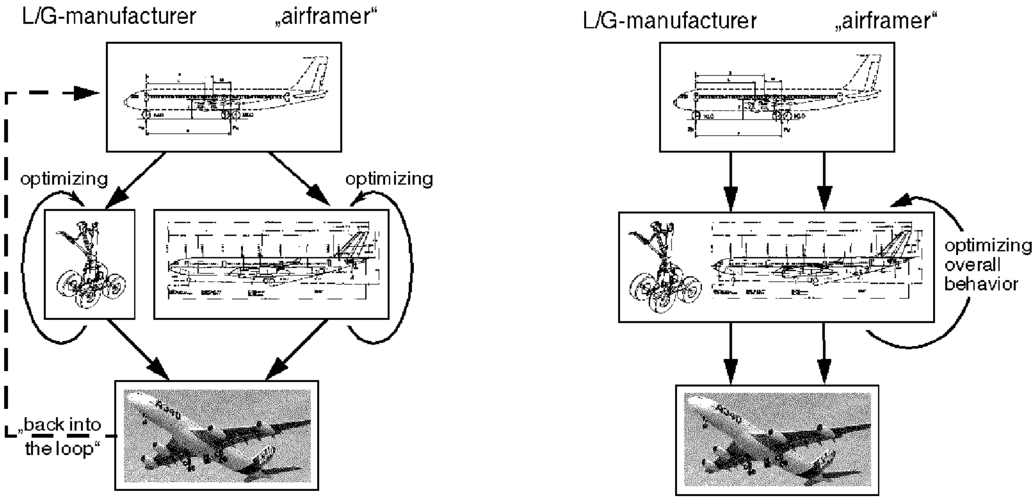


Figure 11: Conventional and Integrated Design Process

Fig. 12 shows an example of such an integrated model for dynamic analysis of airframe / landing gear interaction. A multibody simulation model is assembled, using the interfaces described in Sec. 3. The data comes from a number of different sources – for the airframe, usually a FEA model exists; CAD

models are used to build the simulation model of the landing gears; measurement data enters the simulation for the modeling of the runway and the tires; controllers are imported as code from a control design tool; finally, force laws describing the dynamic behavior of the shock absorbers are programmed and introduced via the programming interface. Simulation runs cover the whole operational envelope on the ground, i.e. touch-down, taxiing, cornering, push-back, and take-off. After the evaluation the results are passed back to the relevant disciplines, most notably forces for stress calculation and certification purposes. Such an integrated design has been performed in the German Flexible Aircraft project [17].

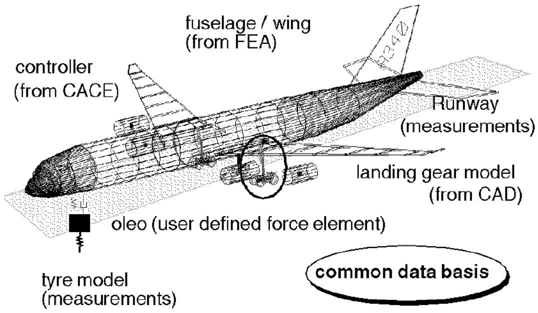


Figure 12: Multidisciplinary simulation model of transport aircraft

5 Summary and Outlook

Numerical simulation is an invaluable tool for the integration of system components. It allows the user to analyze his system up to any chosen degree of complexity, to determine physical variables at any given point of the system, to change design parameters and perform numerical optimizations, and, by doing so, to keep the costs of the vehicle design down.

While the simulation tools have become very sophisticated in their own domains, the simulation of complex systems calls for multidisciplinary simulation. This can be achieved by the coupling of the existing codes. For this purpose, interfaces between the codes have to be developed. These interfaces have to take into consideration the nature of the description of the physical model, numerical properties of the respective simulation methods, and software and hardware implementation issues.

Multibody system analysis is a powerful tool for multidisciplinary analysis of mechanical and mechatronic systems. For the efficient employment of an MBS analysis, a multitude of engineering disciplines have to be considered in the simulation. Combinations of different CAE (computer aided engineering) tools, like MBS, CAD, FEA, and CACE, allow the computation and evaluation of a complex system with the desired accuracy and in affordable computation times. MBS as the analysis tool of the overall system behavior can form the center element of this multidisciplinary design environment.

6 Bibliography

[1] W. Rulka. Effiziente Simulation der Dynamik mechatronischer Systeme für industrielle Anwendungen. Dissertation an der TU Wien. DLR IB 532-01-06. Oberpfaffenhofen, Germany, 2001.

[2] A. Veitl, T. Gordon, A. van de Sand, M. Howell, M. Valasek, O. Vaculin, P. Steinbauer. Methodologies for Coupling Simulation Models and Codes in Mechatronic System Analysis and Design. Veitl, A.; Gordon, T. In: The Dynamics of Vehicles on Roads and Tracks; Supplement to Vehicle System Dynamics, Vol. 33, pp 231-243, 2000.

[3] R. Kübler, W. Schiehlen. Vehicle Modular Simulation in System Dynamics. In: International Conference on Multi-Body Dynamics - New Techniques and Applications, IMechE Conference Transactions 1998-13.

[4] W. Kortüm, W. Rulka, M. Spieck: Simulation of Mechatronic Vehicles with SIMPACK. MOSIS 97, Ostrava, Czech Republic, 1997.

[5] O. Wallrapp. Standardization of Flexible Body Modeling in Multibody System Codes. Part I: Definition of Standard Input Data. Mech. Struc. & Mach., 22(3), 1994, pp. 283 - 304.

[6] O. Vaculin, W.R. Krüger, M. Spieck. Coupling of Multibody and Control Simulation Tools for the Design of Mechatronic Systems. Proceedings of ASME 2001 International Design Engineering Technical Conferences, 9-12. Sept. 2001, Pittsburgh, PA, USA.

[7] W. Kortüm, P. Lugner. Systemdynamik und Regelung von Fahrzeugen. Springer Verlag Berlin, Heidelberg, New-York 1994.

[8] G. Grübel, H.-D. Joos. "Multi-Objective Parameter Synthesis (MOPS)", in: Magni, J.-F., Bennani, S., Terlouw, J. (editors): "Robust Flight Control - A Design Challenge", Lecture Notes in Control and Information Sciences 224, Springer Verlag London Limited, 1997.

[9] H. Wentscher. Design and Analysis of Semi-Active Landing Gears for Transportation Aircraft. DLR-Forschungsbericht 96-11, Köln, 1996.

[10] M. Spieck, W.R. Krüger. Aerodynamic Forces on Elastic Bodies in Multibody Systems. Proceedings of USACM, Sixth U.S. National Congress on Computational Mechanics. August 1-3 2001, Dearborn, Michigan.

[11] R. Ahrem, P. Post, K. Wolf: A Communication Library to Couple Simulation Codes on Distributed Systems for Multi-Physics Computations. In: Proc. of ParCo99 Conference, Imperial College Press, 1999.

[12] W.R. Krüger, R. Heinrich. Strömungs-Struktur-Kopplung von Mehrkörpersimulation und CFD. Proc. of Deutscher Luft- und Raumfahrtkongress 2001, Hamburg, Deutsche Gesellschaft für Luft- und Raumfahrt, 2001.

[13] K. Yi, J.K. Hedrick. Active and Semi-Active Heavy Truck Suspensions to Reduce Pavement Damage. SAE Technical Paper Series, no. 892486, 1989.

[14] D. Cebon. Vehicle Generated Road Damage: A Review. Vehicle System Dynamics, 18, 1989, pp. 107 - 150.

[15] W. Huyer, A. Neumaier. Global optimization by multilevel coordinate search. Journal of Global Optimization, 14, 1999, pp. 331-355.

[16] O. Vaculin, M. Valasek, W. Kortüm. Multi-Objective Semi-Active Truck Suspension by Spatial Decomposition. 17th IAVSD Symposium on Dynamics of Vehicles on Roads and Tracks. Lyngby, DK, 2001. tbp in 2002.

[17] W.R. Krüger, M. Spieck. Interdisciplinary Landing Gear Lay-Out for Large Transport Aircraft. Proceedings of the AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization (AIAA-98-4964), St. Louis, 1998

[18] W.R. Krüger. Integrated Design Process for the Development of Semi-Active Landing Gears for Transport Aircraft. DLR-Forschungsbericht 2001-27, Köln, 2001.